

Announcement for an upcoming Generation of TP-based educational math assistants

Comments on *eduTPS*: “Theorem-Prover based Systems for Education”

Walther Neuper

Institute for Softwaretechnology
Graz University of Technology

Working group *eduTPS* at CADGME'12
June 22 - 24, 2012
Novi Sad, Serbia

This is **NOT** the topic . . .



- 1 Survey on Mathematical Software
Three Examples for TP-based Math Assistants
- 2 Characteristics for a “New Generation”
Conceptual Foundations: Integration of Logics
Technological Features: Transparency, Flexibility
Expected Impact: Education, Research, Development
- 3 Conclusion — Invitation

Survey on Math. Software

Survey

Examples

Characteristics

Foundations

Technology

Impact

Conclusion

In science of ...

SW-tools for ...

... standardized as ..

mathematics

geometry

**CAD/CAM
DGS**

applied sciences

**numerical
computation**

**Spreadsheets
(SSH)**

math. education

**symbolic
computation**

**Computer Algebra
Macsyma 1968**

CAS

**Theorem Proving
Automath 1967**

TPS

Survey on Math. Software

In science of ...

SW-tools for ...

... standardized as ..

mathematics

geometry

CAD/CAM
DGS

applied sciences

**numerical
computation**

Spreadsheets
(SSH)

math. education

symbolic
computation

Computer Algebra
Macsyma 1968

CAS

Theorem Proving
Automath 1967

TPS

Survey on Math. Software

In science of ...

SW-tools for ...

... standardized as ..

mathematics

geometry

**CAD/CAM
DGS**

applied sciences

**numerical
computation**

**Spreadsheets
(SSH)**

math. education

**symbolic
computation**

Computer Algebra
Macsyma 1968

CAS

Theorem Proving
Automath 1967

TPS

Survey on Math. Software

In science of ...

SW-tools for ...

... standardized as ..

mathematics

geometry

**CAD/CAM
DGS**

applied sciences

**numerical
computation**

**Spreadsheets
(SSH)**

math. education

**symbolic
computation**

Computer Algebra
Macsyma 1968

CAS

Theorem Proving (TP)
Automath 1967

TPS

Survey on Math. Software

In science of ...

SW-tools for ...

... standardized as ..

mathematics

geometry

**CAD/CAM
DGS**

applied sciences

**numerical
computation**

**Spreadsheets
(SSH)**

math. education

**symbolic
computation**

Computer Algebra
Macsyma **1968**

CAS

Theorem Proving (TP)
Automath **1967**

TPS

Survey on Math. Software

In science of ...

SW-tools for ...

... standardized as ..

mathematics

geometry

CAD/CAM
DGS

applied sciences

numerical
computation

Spreadsheets
(SSH)

math. education

symbolic
computation

Computer Algebra
Macsyma 1968

CAS

Theorem Proving
Automath 1967

TPS

Survey on Math. Software

In science of ...

SW-tools for ...

... standardized as ..

mathematics

geometry

CAD/CAM
DGS

applied sciences

numerical
computation

Spreadsheets
(SSH)

math. education

symbolic
computation

Computer Algebra
Macsyma 1968

CAS

Theorem Proving
Automath 1967

TPS

Survey on Math. Software

In science of ...

SW-tools for ...

... standardized as ..

mathematics

geometry

CAD/CAM
DGS

applied sciences

numerical
computation

Spreadsheets
(SSH)

math. education

symbolic
computation

Computer Algebra
Macsyma 1968

CAS

Theorem Proving
Automath 1967

TPS

Survey on Math. Software

In science of ...

SW-tools for ...

... standardized as ..

mathematics

geometry

CAD/CAM
DGS

applied sciences

numerical
computation

Spreadsheets
(SSH)

math. education

symbolic
computation

Computer Algebra
Macsyma 1968

CAS

Theorem Proving
Automath 1967

TPS
TP-based Systems

Survey on Math. Software

In science of ...

SW-tools for ...

... standardized as ..

mathematics

geometry

CAD/CAM
DGS

applied sciences

numerical
computation

Spreadsheets
(SSH)

math. education

symbolic
computation

Computer Algebra
Macsyma 1968

CAS

Theorem Proving
Automath 1967

TPS

Survey on Math. Software

In science of ...

SW-tools for ...

... standardized as ..

mathematics

geometry

CAD/CAM
DGS

applied sciences

numerical
computation

Spreadsheets
(SSH)

math. education

symbolic
computation

Computer Algebra
Macsyma 1968

CAS

Theorem Proving
Automath 1967

TPS

Survey on Math. Software

In science of ...

SW-tools for ...

... standardized as ..

mathematics

geometry

CAD/CAM
DGS

applied sciences

numerical
computation

Spreadsheets
(SSH)

math. education

symbolic
computation

Computer Algebra
Macsyma 1968

CAS

Theorem Proving
Automath 1967

TPS !?!

- 1 Survey on Mathematical Software
Three Examples for TP-based Math Assistants
- 2 Characteristics for a “New Generation”
Conceptual Foundations: Integration of Logics
Technological Features: Transparency, Flexibility
Expected Impact: Education, Research, Development
- 3 Conclusion — Invitation

The screenshot displays the WinGCLC software interface with a window titled "WinGCLC - [triangle.gcl]". The interface is divided into a code editor on the left and a graphical workspace on the right. The code editor contains the following text:

```
point A 5 20
point B 45 20
point C 10 40

% Draw triangle points
cmark_lt A
cmark_rt B
cmark_lt C

% Draw triangle sides
drawsegment A B
drawsegment B C
drawsegment C A

% Construct perpendicular bisectors
sed a C B
sed b A C
sed c A B

% Draw perpendicular bisectors
drawdashline a
drawdashline b
drawdashline c

intersec O_1 a b
cmark_lt O_1
intersec O_2 a c
cmark_rt O_2

drawcircle O_1 A

prove {identical O_1 O_2}
```

Below the code editor, a status bar indicates "The conjecture successfully proved." The graphical workspace shows a circle with an inscribed triangle ABC. Dashed lines represent the perpendicular bisectors of the triangle's sides, which intersect at a central point labeled O_1. Another point O_2 is also marked, and the software is proving that O_1 and O_2 are identical. The status bar at the bottom of the window shows coordinates: "lin 10, Col 1", "x=12.00", "y=22.30", "Zoom=3.58", and "GMR".

GeoGebra's "academic relative", see
prove {identical O_1 O_2} at bottom left.

Socos, Abo Akademi Turku

The screenshot shows the Eclipse IDE interface. The main editor displays a program in a language that appears to be a form of pseudocode or a simple imperative language. The program is as follows:

```
max [ a:array[n,int]
      m: result int ]
k: pvar int

n:=0
k:=1
m:=0

loop
  0<=k and k<=n
  0<=m and m<k
  forall (i:int): 0<=i and i<k => a(i) <= a(m)

  [k<n]
  n:=0
  0<=m and m<n
  forall (i:int): 0<=i and i<n => a(i) <= a(m)

  [a(k)>=a(m)]
  k:=k+1
  m:=m
  decreasing
```

The flowchart below the code illustrates the execution flow. It starts with an initialization block where $n=0$, $k=1$, and $m=0$. A loop block contains the conditions $0 \leq k \text{ and } k \leq n$ and $0 \leq m \text{ and } m < k$, followed by a forall loop: $\text{forall } (i:\text{int}): 0 \leq i \text{ and } i < k \Rightarrow a(i) \leq a(m)$. A branch labeled $[k < n]$ leads to a block where $n=0$, $0 \leq m \text{ and } m < n$, and another forall loop: $\text{forall } (i:\text{int}): 0 \leq i \text{ and } i < n \Rightarrow a(i) \leq a(m)$. From there, a branch labeled $[a(k) \geq a(m)]$ leads to a block where $k := k + 1$, $m := m$, and a 'decreasing' annotation. The flowchart also shows a return path from the loop back to the start of the loop.

The console window at the bottom shows two error messages:

```
2 errors, 0 warnings, 0 others
Description
Errors (2 items)
[-1] a(k) > a(m)
[-2] k < n
[-3] 0 < k AND k <= n AND 0 <= m AND m < k
AND FORALL (i: int): 0 <= i AND i < k => a(i) <= a(m)
-----
[1] 0 <= k + 1 AND k + 1 < k + 1
[-1] a(k) > a(m)
[-2] k < n
[-3] 0 < k AND k <= n AND 0 <= m AND m < k
AND FORALL (i: int): 0 <= i AND i < k => a(i) <= a(m)
-----
[1] FORALL (i: int): 0 <= i AND i < k + 1 => a(i) <= a(k + 1)
```

For software “correct by construction” in education.

The screenshot shows the ISAC software interface with the following components:

- Problem Browser:** Lists the problem as "Problem (Biegelinie.thy, [Biegelinien])" and "Problem (Biegelinie.thy, [vonBelastungZu, Biegelinien])".
- Worksheet:** Contains a step-by-step solution:
 - Problem (Biegelinie.thy, [Biegelinien])
 - Problem (Biegelinie.thy, [vonBelastungZu, Biegelinien])
 - $q \ x = q \ 0$
 - $-q \ x = -q \ 0$
 - $Q' \ x = -q \ 0$
 - Integrate (- q 0, x)
 - $Q \ x = \text{Integral} -q \ 0 \ D \ x$
 - $Q \ x = \text{Integral} -1 \ * \ q \ 0 \ D \ x$
 - $Q \ x = c + -1 \ * \ q \ 0 \ * \ x$
- Text Area:** Contains explanatory text:
 - $\frac{dV}{dx} = -g$ also
 - und folglich $\frac{dV}{dx}$ also
 - sfunktion ist gleich der 1.
 - n der Querkraftlinie.
 - Neigung der
 - Höhe der Belastung
 - ungen hinsichtlich der
 - $f + dM) = 0$
 - alich kleine Größe höherer
 - gt werden
 - $M' = V$
 - und somit $\frac{dM}{dx}$. Die 1. Ableitung
 - der Funktion der Momentenlinie liefert die
 - Querkraft für jede Stelle.

For step-wise problem solving in applied mathematics.

Outline

- 1 Survey on Mathematical Software
Three Examples for TP-based Math Assistants
- 2 Characteristics for a “New Generation”
Conceptual Foundations: Integration of Logics
Technological Features: Transparency, Flexibility
Expected Impact: Education, Research, Development
- 3 Conclusion — Invitation

Conceptual Foundations

Mathematics is the science of *reasoning* ...

- each operation can be *proved*
- ... to “*prove*” is the essence which distinguishes math.

TP ((Computer) Theorem *Proving*) realises this essence.

Consequences for TP-based software:

- 1 TP provides a logical framework for CAS, DGS, ...
- 2 TP is **integrative** (rather than competitive)
- 3 TP covers an essential range of mathematics including STEM¹.

¹STEM: “Science, Technology, Engineering and Mathematics”

Conceptual Foundations

Mathematics is the science of *reasoning* ...

- each operation can be *proved*
- ... to “*prove*” is the essence which distinguishes math.

TP ((Computer) Theorem *Proving*) realises this essence.

Consequences for TP-based software:

- 1 TP provides a logical framework for CAS, DGS, ...
- 2 TP is **integrative** (rather than competitive)
- 3 TP covers an essential range of mathematics including STEM¹.

¹STEM: “Science, Technology, Engineering and Mathematics”

Conceptual Foundations

Mathematics is the science of *reasoning* ...

- each operation can be *proved*
- ... to “*prove*” is the essence which distinguishes math.

TP ((Computer) Theorem *Proving*) realises this essence.

Consequences for TP-based software:

- 1 TP provides a logical framework for CAS, DGS, ...
- 2 TP is **integrative** (rather than competitive)
- 3 TP covers an essential range of mathematics including STEM¹.

¹STEM: “Science, Technology, Engineering and Mathematics”

7 fundamental capabilities

in PISA's competence model for mathematics:

- 1 **Communication:** *"...perceiving the existence of some challenge and recognizing a problem situation ..."*
- 2 **Mathematising:** *"...transforming a problem defined in the real world to a strictly mathematical form ..."*
- 3 **Representation:** *"...selecting, interpreting and using a variety of representations to capture a situation ..."*
- 4 **Reasoning** and argument: *"...logically rooted thought processes that check a justification that is given, ..."*
- 5 Devising **strategies** for solving problems: *"...critical control processes that solve problems ..."*
- 6 Using **symbolic**, formal and technical language and **operations:** *"...within a mathematical context ..."*
- 7 Using mathematical **tools:** *"...being able to make use of various tools that may assist math activity ..."*

7 fundamental capabilities

in PISA's competence model for mathematics:

- 1 **Communication:** "... *perceiving the existence of some challenge and recognizing a problem situation ...* "
- 2 **Mathematising:** "... *transforming a problem defined in the real world to a strictly mathematical form ...* "
- 3 **Representation:** "... *selecting, interpreting and using a variety of representations to capture a situation ...* "
- 4 **Reasoning** and argument: "... *logically rooted thought processes that check a justification that is given, ...* "
- 5 Devising **strategies** for solving problems: "... *critical control processes that solve problems ...* "
- 6 Using **symbolic**, formal and technical language and **operations:** "... *within a mathematical context ...* "
- 7 **Using mathematical tools:** "... *being able to make use of various tools that may assist math activity ...* "

7 fundamental capabilities . . .

. . . respectively **covered by TP-based systems:**

- 1 **Communication:** not *specifically* addressed by TP
- 2 **Mathematising:** **formalisation** is a prerequisite for TPS support — *can be prepared and hidden from student*
- 3 **Representation:** various **specifications** can be offered — *and tried out using next-step-guidance*
- 4 **Reasoning:** every operation in TPS has a **mechanized justification** — *can be hidden and handled on request*
- 5 **Strategies:** various solving **algorithms** can be offered — *and tried out using next-step-guidance*
- 6 **Symbolic operations:** **all** TPS operations have a **symbolic** representation — *next-step-guidance helps.*
- 7 **Tools:** TPS address the other capabilities above

Doesn't all that overstrain students ? *Not necessarily !*

7 fundamental capabilities . . .

. . . respectively **covered by TP-based systems**:

- 1 Communication: not *specifically* addressed by TP
- 2 Mathematizing: **formalisation** is a prerequisite for TPS support — *can be prepared and hidden from student*
- 3 Representation: various **specifications** can be offered — *and tried out using next-step-guidance*
- 4 Reasoning: every operation in TPS has a **mechanized justification** — *can be hidden and handled on request*
- 5 Strategies: various solving **algorithms** can be offered — *and tried out using next-step-guidance*
- 6 Symbolic operations: **all** TPS operations have a **symbolic** representation — *next-step-guidance helps.*
- 7 Tools: TPS address the other capabilities above

Doesn't all that overstrain students ? *Not necessarily !*

7 fundamental capabilities . . .

. . . respectively **covered by TP-based systems:**

- 1 Communication: not *specifically* addressed by TP
- 2 Mathematizing: **formalisation** is a prerequisite for TPS support — *can be prepared and hidden from student*
- 3 Representation: various **specifications** can be offered — *and tried out using next-step-guidance*
- 4 Reasoning: every operation in TPS has a **mechanized justification** — *can be hidden and handled on request*
- 5 Strategies: various solving **algorithms** can be offered — *and tried out using next-step-guidance*
- 6 Symbolic operations: **all** TPS operations have a **symbolic** representation — *next-step-guidance helps.*
- 7 Tools: TPS address the other capabilities above

Doesn't all that overstrain students ? *Not necessarily !*

7 fundamental capabilities . . .

. . . respectively **covered by TP-based systems**:

- 1 Communication: not *specifically* addressed by TP
- 2 Mathematising: **formalisation** is a prerequisite for TPS support — can be prepared and hidden from student
- 3 Representation: various **specifications** can be offered — and tried out using next-step-guidance
- 4 Reasoning: every operation in TPS has a **mechanized justification** — can be hidden and handled on request
- 5 Strategies: various solving **algorithms** can be offered — and tried out using next-step-guidance
- 6 Symbolic operations: **all** TPS operations have a **symbolic** representation — next-step-guidance helps.
- 7 Tools: TPS address the other capabilities above

Doesn't all that overstrain students ? Not necessarily !

7 fundamental capabilities . . .

. . . respectively **covered by TP-based systems**:

- 1 Communication: not *specifically* addressed by TP
- 2 Mathematising: **formalisation** is a prerequisite for TPS support — *can be prepared and hidden from student*
- 3 Representation: various **specifications** can be offered — *and tried out using next-step-guidance*
- 4 Reasoning: every operation in TPS has a **mechanized justification** — *can be hidden and handled on request*
- 5 Strategies: various solving **algorithms** can be offered — *and tried out using next-step-guidance*
- 6 Symbolic operations: **all** TPS operations have a **symbolic** representation — *next-step-guidance helps.*
- 7 Tools: TPS address the other capabilities above

Doesn't all that overstrain students ? *Not necessarily !*

7 fundamental capabilities . . .

. . . respectively **covered by TP-based systems**:

- 1 Communication: not *specifically* addressed by TP
- 2 Mathematizing: **formalisation** is a prerequisite for TPS support — can be prepared and hidden from student
- 3 Representation: various **specifications** can be offered — and tried out using next-step-guidance
- 4 Reasoning: every operation in TPS has a **mechanized justification** — can be hidden and handled on request
- 5 Strategies: various solving **algorithms** can be offered — and tried out using next-step-guidance
- 6 Symbolic operations: **all** TPS operations have a **symbolic** representation — next-step-guidance helps.
- 7 Tools: TPS address the other capabilities above

Doesn't all that overstrain students ? Not necessarily !

7 fundamental capabilities . . .

. . . respectively **covered by TP-based systems**:

- 1 Communication: not *specifically* addressed by TP
- 2 Mathematising: **formalisation** is a prerequisite for TPS support — *can be prepared and hidden from student*
- 3 Representation: various **specifications** can be offered — *and tried out using next-step-guidance*
- 4 Reasoning: every operation in TPS has a **mechanized justification** — *can be hidden and handled on request*
- 5 Strategies: various solving **algorithms** can be offered — *and tried out using next-step-guidance*
- 6 Symbolic operations: **all** TPS operations have a **symbolic** representation — *next-step-guidance helps.*
- 7 Tools: TPS address the other capabilities above

Doesn't all that overstrain students ? *Not necessarily !*

7 fundamental capabilities . . .

. . . respectively **covered by TP-based systems**:

- 1 Communication: not *specifically* addressed by TP
- 2 Mathematising: **formalisation** is a prerequisite for TPS support — *can be prepared and hidden from student*
- 3 Representation: various **specifications** can be offered — *and tried out using next-step-guidance*
- 4 Reasoning: every operation in TPS has a **mechanized justification** — *can be hidden and handled on request*
- 5 Strategies: various solving **algorithms** can be offered — *and tried out using next-step-guidance*
- 6 Symbolic operations: **all** TPS operations have a **symbolic** representation — *next-step-guidance helps.*
- 7 Tools: TPS address the other capabilities above

Doesn't all that overstrain students ? *Not necessarily !*

7 fundamental capabilities ...

... respectively **covered by TP-based systems**:

- 1 Communication: not *specifically* addressed by TP
- 2 Mathematising: **formalisation** is a prerequisite for TPS support — *can be prepared and hidden from student*
- 3 Representation: various **specifications** can be offered — *and tried out using next-step-guidance*
- 4 Reasoning: every operation in TPS has a **mechanized justification** — *can be hidden and handled on request*
- 5 Strategies: various solving **algorithms** can be offered — *and tried out using next-step-guidance*
- 6 Symbolic operations: **all** TPS operations have a **symbolic** representation — *next-step-guidance helps.*
- 7 Tools: TPS address the other capabilities above

Doesn't all that overstrain students ? *Not necessarily !*

7 fundamental capabilities . . .

. . . respectively **covered by TP-based systems**:

- 1 Communication: not *specifically* addressed by TP
- 2 Mathematizing: **formalisation** is a prerequisite for TPS support — *can be prepared and hidden from student*
- 3 Representation: various **specifications** can be offered — *and tried out using next-step-guidance*
- 4 Reasoning: every operation in TPS has a **mechanized justification** — *can be hidden and handled on request*
- 5 Strategies: various solving **algorithms** can be offered — *and tried out using next-step-guidance*
- 6 Symbolic operations: **all** TPS operations have a **symbolic** representation — *next-step-guidance helps.*
- 7 Tools: TPS address the other capabilities above

Doesn't all that overstrain students ? **Not necessarily !**

7 fundamental capabilities ...

... respectively **covered by TP-based systems**:

- 1 Communication: not *specifically* addressed by TP
- 2 Mathematising: **formalisation** is a prerequisite for TPS support — **can be prepared and hidden from student**
- 3 Representation: various **specifications** can be offered — **and tried out using next-step-guidance**
- 4 Reasoning: every operation in TPS has a **mechanized justification** — **can be hidden and handled on request**
- 5 Strategies: various solving **algorithms** can be offered — **and tried out using next-step-guidance**
- 6 Symbolic operations: **all** TPS operations have a **symbolic** representation — **next-step-guidance helps.**
- 7 Tools: TPS address the other capabilities above

Doesn't all that overstrain students ? Not necessarily !

7 fundamental capabilities ...

... respectively **covered by TP-based systems**:

- 1 Communication: not *specifically* addressed by TP
- 2 Mathematising: **formalisation** is a prerequisite for TPS support — **can be prepared and hidden from student**
- 3 Representation: various **specifications** can be offered — **and tried out using next-step-guidance**
- 4 Reasoning: every operation in TPS has a **mechanized justification** — **can be hidden and handled on request**
- 5 Strategies: various solving **algorithms** can be offered — **and tried out using next-step-guidance**
- 6 Symbolic operations: **all** TPS operations have a **symbolic** representation — **next-step-guidance helps.**
- 7 Tools: TPS address the other capabilities above

Doesn't all that overstrain students ? Not necessarily !

7 fundamental capabilities ...

... respectively **covered by TP-based systems**:

- 1 Communication: not *specifically* addressed by TP
- 2 Mathematising: **formalisation** is a prerequisite for TPS support — **can be prepared and hidden from student**
- 3 Representation: various **specifications** can be offered — **and tried out using next-step-guidance**
- 4 Reasoning: every operation in TPS has a **mechanized justification** — **can be hidden and handled on request**
- 5 Strategies: various solving **algorithms** can be offered — **and tried out using next-step-guidance**
- 6 Symbolic operations: **all** TPS operations have a **symbolic** representation — **next-step-guidance helps.**
- 7 Tools: TPS address the other capabilities above

Doesn't all that overstrain students ? Not necessarily !

7 fundamental capabilities ...

... respectively **covered by TP-based systems**:

- 1 Communication: not *specifically* addressed by TP
- 2 Mathematising: **formalisation** is a prerequisite for TPS support — **can be prepared and hidden from student**
- 3 Representation: various **specifications** can be offered — **and tried out using next-step-guidance**
- 4 Reasoning: every operation in TPS has a **mechanized justification** — **can be hidden and handled on request**
- 5 Strategies: various solving **algorithms** can be offered — **and tried out using next-step-guidance**
- 6 Symbolic operations: **all** TPS operations have a **symbolic** representation — *next-step-guidance helps.*
- 7 Tools: TPS address the other capabilities above

Doesn't all that overstrain students ? **Not necessarily !**

7 fundamental capabilities ...

... respectively **covered by TP-based systems**:

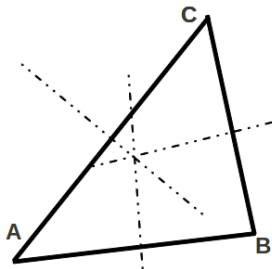
- 1 Communication: not *specifically* addressed by TP
- 2 Mathematising: **formalisation** is a prerequisite for TPS support — **can be prepared and hidden from student**
- 3 Representation: various **specifications** can be offered — **and tried out using next-step-guidance**
- 4 Reasoning: every operation in TPS has a **mechanized justification** — **can be hidden and handled on request**
- 5 Strategies: various solving **algorithms** can be offered — **and tried out using next-step-guidance**
- 6 Symbolic operations: **all** TPS operations have a **symbolic** representation — **next-step-guidance helps.**
- 7 Tools: TPS address the other capabilities above

Doesn't all that overstrain students ? Not necessarily !

For example “Mathematising”

*The perpendicular midlines of the sides
in a triangle meet in one point.*

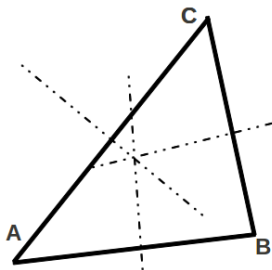
We know !



For example “Mathematising”

*The perpendicular midlines of the sides
in a triangle meet in one point.*

We know !



For example “Mathematising”

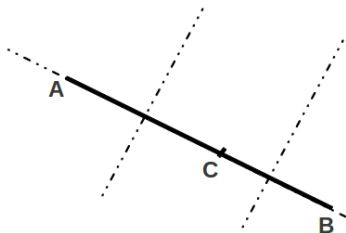
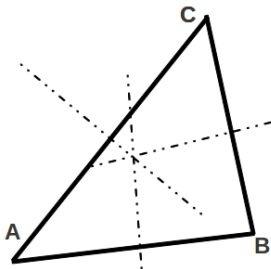
*The perpendicular midlines of the sides
in a triangle meet in one point.*

We know !

Really?

prove {identical O_1 O_2 } requires
“non-degeneracy conditions”

Without “mathematising” (specifying formally) GCLC cannot *prove!*



For example “Mathematising”

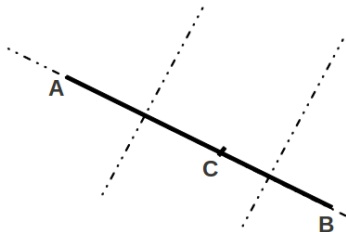
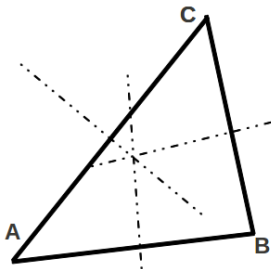
*The perpendicular midlines of the sides
in a triangle meet in one point.*

We know !

Really?

prove {identical $O_1 O_2$ } requires
“non-degeneracy conditions”

Without “mathematising” (specifying formally) GCLC cannot *prove!*



For example “Mathematising”

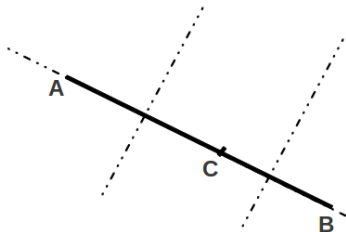
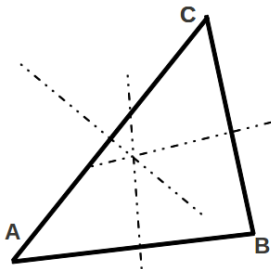
*The perpendicular midlines of the sides
in a triangle meet in one point.*

We know !

Really?

prove {identical $O_1 O_2$ } requires
“non-degeneracy conditions”

Without “mathematising” (specifying formally) GCLC **cannot prove!**



Outline

- 1 Survey on Mathematical Software
Three Examples for TP-based Math Assistants
- 2 Characteristics for a “New Generation”
Conceptual Foundations: Integration of Logics
Technological Features: Transparency, Flexibility
Expected Impact: Education, Research, Development
- 3 Conclusion — Invitation

Technological Features

These features arise from TP ...

- 1 **check user input** automatically, **flexibly** and reliably:
Input establishes a *proof situation* (for *automated* proving)
with respect to the logical context
- 2 **give explanations** on request by learners:
All underlying mathematics knowledge is **transparent** due to
the “LCF-paradigm” (not a program code!)
- 3 **propose a next step** if learners get stuck:
“next-step-guidance” due to Lucas-Interpretation.

Thus featuring software support for:

- **step-wise solving math** problems in STEM
- learning interactively like with a **chess-program**
- ...

Technological Features

These features arise from TP ...

- 1 **check user input** automatically, **flexibly** and reliably:
Input establishes a *proof situation* (for *automated* proving) with respect to the logical context
- 2 **give explanations** on request by learners:
All underlying mathematics knowledge is **transparent** due to the “LCF-paradigm” (not a program code!)
- 3 **propose a next step** if learners get stuck:
“next-step-guidance” due to Lucas-Interpretation.

Thus featuring software support for:

- **step-wise solving math** problems in STEM
- learning interactively like with a **chess-program**
- ...

Technological Features

These features arise from TP ...

- 1 **check user input** automatically, **flexibly** and reliably:
Input establishes a *proof situation* (for *automated* proving)
with respect to the logical context
- 2 **give explanations** on request by learners:
All underlying mathematics knowledge is **transparent** due to
the “LCF-paradigm” (not a program code!)
- 3 **propose a next step** if learners get stuck:
“next-step-guidance” due to Lucas-Interpretation.

Thus featuring software support for:

- **step-wise solving math** problems in STEM
- learning interactively like with a **chess-program**
- ...

Technological Features

These features arise from TP ...

- 1 **check user input** automatically, **flexibly** and reliably:
Input establishes a *proof situation* (for *automated* proving)
with respect to the logical context
- 2 **give explanations** on request by learners:
All underlying mathematics knowledge is **transparent** due to
the “LCF-paradigm” (not a program code!)
- 3 **propose a next step** if learners get stuck:
“next-step-guidance” due to Lucas-Interpretation.

Thus featuring software support for:

- **step-wise solving math** problems in STEM
- learning interactively like with a **chess-program**
- ...

Technological Features

These features arise from TP ...

- 1 **check user input** automatically, **flexibly** and reliably:
Input establishes a *proof situation* (for *automated* proving) with respect to the logical context
- 2 **give explanations** on request by learners:
All underlying mathematics knowledge is **transparent** due to the “LCF-paradigm” (not a program code!)
- 3 **propose a next step** if learners get stuck:
“next-step-guidance” due to Lucas-Interpretation.

Thus featuring software support for:

- **step-wise solving math** problems in STEM
- learning interactively like with a **chess-program**
- ...

Technological Features

These features arise from TP ...

- 1 **check user input** automatically, **flexibly** and reliably:
Input establishes a *proof situation* (for *automated* proving) with respect to the logical context
- 2 **give explanations** on request by learners:
All underlying mathematics knowledge is **transparent** due to the “LCF-paradigm” (not a program code!)
- 3 **propose a next step** if learners get stuck:
“next-step-guidance” due to Lucas-Interpretation.

Thus featuring software support for:

- **step-wise solving math** problems in STEM
- learning interactively like with a **chess-program**
- ...

Technological Features

These features arise from TP ...

- 1 **check user input** automatically, **flexibly** and reliably:
Input establishes a *proof situation* (for *automated* proving) with respect to the logical context
- 2 **give explanations** on request by learners:
All underlying mathematics knowledge is **transparent** due to the “LCF-paradigm” (not a program code!)
- 3 **propose a next step** if learners get stuck:
“next-step-guidance” due to Lucas-Interpretation.

Thus featuring software support for:

- **step-wise solving math** problems in STEM
- learning interactively like with a **chess-program**
- ...

Technological Features

These features arise from TP ...

- 1 **check user input** automatically, **flexibly** and reliably:
Input establishes a *proof situation* (for *automated* proving) with respect to the logical context
- 2 **give explanations** on request by learners:
All underlying mathematics knowledge is **transparent** due to the “LCF-paradigm” (not a program code!)
- 3 **propose a next step** if learners get stuck:
“next-step-guidance” due to Lucas-Interpretation.

Thus featuring software support for:

- **step-wise solving math** problems in STEM
- learning interactively like with a **chess-program**
- ...

Technological Features

These features arise from TP ...

- 1 **check user input** automatically, **flexibly** and reliably:
Input establishes a *proof situation* (for *automated* proving) with respect to the logical context
- 2 **give explanations** on request by learners:
All underlying mathematics knowledge is **transparent** due to the “LCF-paradigm” (not a program code!)
- 3 **propose a next step** if learners get stuck:
“next-step-guidance” due to Lucas-Interpretation.

Thus featuring software support for:

- **step-wise solving math** problems in STEM
- learning interactively like with a **chess-program**
- ...

Technological Features

These features arise from TP ...

- 1 **check user input** automatically, **flexibly** and reliably:
Input establishes a *proof situation* (for *automated* proving) with respect to the logical context
- 2 **give explanations** on request by learners:
All underlying mathematics knowledge is **transparent** due to the “LCF-paradigm” (not a program code!)
- 3 **propose a next step** if learners get stuck:
“next-step-guidance” due to Lucas-Interpretation.

Thus featuring software support for:

- **step-wise solving math** problems in STEM
- learning interactively like with a **chess-program**
- ...

Outline

- 1 Survey on Mathematical Software
Three Examples for TP-based Math Assistants
- 2 Characteristics for a “New Generation”
Conceptual Foundations: Integration of Logics
Technological Features: Transparency, Flexibility
Expected Impact: Education, Research, Development
- 3 Conclusion — Invitation

Impact expected . . .

- for **learners** in STEM from high-school to university:
 - *independent learning* in all phases of problem solving
 - construction of solutions with “*next-step-guidance*”
 - *flexible* access to knowledge in context of steps
- for **researchers** in science education, cognitive science
 - *logging of steps* for analysis of problem solving behav.
 - summative *assessment* of step-wise problem solving
 - mechanised *analysis of prerequisites* in curricula
- for **educational planners** and administrators
 - curriculum development on *mechanised knowledge*
 - *cross-institutional interfaces* are explicit
 - *summative assessment* of institutions
- for **developers** of systems, knowledge and dialogs
 - TP components contribute *support by automation* in checking user input, access to knowledge, . . .
 - *but* require adherence to TP standards and logics

Impact expected . . .

- for **learners** in STEM from high-school to university:
 - *independent learning* in all phases of problem solving
 - construction of solutions with “*next-step-guidance*”
 - *flexible* access to knowledge in context of steps
- for **researchers** in science education, cognitive science
 - *logging of steps* for analysis of problem solving behav.
 - summative *assessment* of step-wise problem solving
 - mechanised *analysis of prerequisites* in curricula
- for **educational planners** and administrators
 - curriculum development on *mechanised knowledge*
 - *cross-institutional interfaces* are explicit
 - *summative assessment* of institutions
- for **developers** of systems, knowledge and dialogs
 - TP components contribute *support by automation* in checking user input, access to knowledge, . . .
 - *but* require adherence to TP standards and logics

Impact expected . . .

- for **learners** in STEM from high-school to university:
 - *independent learning* in all phases of problem solving
 - construction of solutions with “*next-step-guidance*”
 - *flexible* access to knowledge in context of steps
- for **researchers** in science education, cognitive science
 - *logging of steps* for analysis of problem solving behav.
 - summative *assessment* of step-wise problem solving
 - mechanised *analysis of prerequisites* in curricula
- for **educational planners** and administrators
 - curriculum development on *mechanised knowledge*
 - *cross-institutional interfaces* are explicit
 - *summative assessment* of institutions
- for **developers** of systems, knowledge and dialogs
 - TP components contribute *support by automation* in checking user input, access to knowledge, . . .
 - *but* require adherence to TP standards and logics

Impact expected . . .

- for **learners** in STEM from high-school to university:
 - *independent learning* in all phases of problem solving
 - construction of solutions with “*next-step-guidance*”
 - *flexible* access to knowledge in context of steps
- for **researchers** in science education, cognitive science
 - *logging of steps* for analysis of problem solving behav.
 - summative *assessment* of step-wise problem solving
 - mechanised *analysis of prerequisites* in curricula
- for **educational planners** and administrators
 - curriculum development on *mechanised knowledge*
 - *cross-institutional interfaces* are explicit
 - *summative assessment* of institutions
- for **developers** of systems, knowledge and dialogs
 - TP components contribute *support by automation* in checking user input, access to knowledge, . . .
 - *but* require adherence to TP standards and logics

Impact expected . . .

- for **learners** in STEM from high-school to university:
 - *independent learning* in all phases of problem solving
 - construction of solutions with “*next-step-guidance*”
 - *flexible* access to knowledge in context of steps
- for **researchers** in science education, cognitive science
 - *logging of steps* for analysis of problem solving behav.
 - summative *assessment* of step-wise problem solving
 - mechanised *analysis of prerequisites* in curricula
- for **educational planners** and administrators
 - curriculum development on *mechanised knowledge*
 - *cross-institutional interfaces* are explicit
 - *summative assessment* of institutions
- for **developers** of systems, knowledge and dialogs
 - TP components contribute *support by automation* in checking user input, access to knowledge, . . .
 - *but* require adherence to TP standards and logics

Impact expected . . .

- for **learners** in STEM from high-school to university:
 - *independent learning* in all phases of problem solving
 - construction of solutions with “*next-step-guidance*”
 - *flexible* access to knowledge in context of steps
- for **researchers** in science education, cognitive science
 - *logging of steps* for analysis of problem solving behav.
 - summative *assessment* of step-wise problem solving
 - mechanised *analysis of prerequisites* in curricula
- for **educational planners** and administrators
 - curriculum development on *mechanised knowledge*
 - *cross-institutional interfaces* are explicit
 - *summative assessment* of institutions
- for **developers** of systems, knowledge and dialogs
 - TP components contribute *support by automation* in checking user input, access to knowledge, . . .
 - *but* require adherence to TP standards and logics

Impact expected . . .

- for **learners** in STEM from high-school to university:
 - *independent learning* in all phases of problem solving
 - construction of solutions with “*next-step-guidance*”
 - *flexible* access to knowledge in context of steps
- for **researchers** in science education, cognitive science
 - *logging of steps* for analysis of problem solving behav.
 - summative *assessment* of step-wise problem solving
 - mechanised *analysis of prerequisites* in curricula
- for **educational planners** and administrators
 - curriculum development on *mechanised knowledge*
 - *cross-institutional interfaces* are explicit
 - *summative assessment* of institutions
- for **developers** of systems, knowledge and dialogs
 - TP components contribute *support by automation* in checking user input, access to knowledge, . . .
 - *but* require adherence to TP standards and logics

Impact expected . . .

- for **learners** in STEM from high-school to university:
 - *independent learning* in all phases of problem solving
 - construction of solutions with “*next-step-guidance*”
 - *flexible* access to knowledge in context of steps
- for **researchers** in science education, cognitive science
 - *logging of steps* for analysis of problem solving behav.
 - summative *assessment* of step-wise problem solving
 - mechanised *analysis of prerequisites* in curricula
- for **educational planners** and administrators
 - curriculum development on *mechanised knowledge*
 - *cross-institutional interfaces* are explicit
 - *summative assessment* of institutions
- for **developers** of systems, knowledge and dialogs
 - TP components contribute *support by automation* in checking user input, access to knowledge, . . .
 - *but* require adherence to TP standards and logics

Outline

- 1 Survey on Mathematical Software
Three Examples for TP-based Math Assistants
- 2 Characteristics for a “New Generation”
Conceptual Foundations: Integration of Logics
Technological Features: Transparency, Flexibility
Expected Impact: Education, Research, Development
- 3 Conclusion — Invitation

Conclusion

Renewed Pedagogy for the Future



STEM

individualise,

meaningful math

inquiry-based learning

embodiment, social experience

Effective Practice in Math Education

infinite variety of human thought

distilled to science of math

abstracted to logics

implemented in

software

TP



Mechanised Foundation of Math

Let's hope for **joint inspiration** how to relate the sides !

Thank you for listening to this invitation !

Conclusion

Renewed Pedagogy for the Future



STEM

individualise,

meaningful math

inquiry-based learning

embodiment, social experience

Effective Practice in Math Education

infinite variety of human thought

distilled to science of math

abstracted to logics

implemented in

software

TP



Mechanised Foundation of Math

Let's hope for **joint inspiration** how to relate the sides !

Thank you for listening to this invitation !

Conclusion

Renewed Pedagogy for the Future



STEM

individualise,

meaningful math

inquiry-based learning

embodiment, social experience

Effective Practice in Math Education

**infinite variety of human thought
distilled to science of math**

abstracted to logics

implemented in

software

TP



=====
Mechanised Foundation of Math

Let's hope for **joint inspiration** how to relate the sides !

Thank you for listening to this invitation !

Conclusion

Renewed Pedagogy for the Future



STEM

individualise,

meaningful math

inquiry-based learning

embodiment, social experience

Effective Practice in Math Education

infinite variety of human thought
distilled to science of math
abstracted to logics

implemented in

software

TP



Mechanised Foundation of Math

Let's hope for **joint inspiration** how to relate the sides !

Thank you for listening to this invitation !

Conclusion

Renewed Pedagogy for the Future



STEM

individualise,

meaningful math

inquiry-based learning

embodiment, social experience

Effective Practice in Math Education

infinite variety of human thought
distilled to science of math
abstracted to logics
implemented in

software

TP



Mechanised Foundation of Math

Let's hope for **joint inspiration** how to relate the sides !

Thank you for listening to this invitation !

infinite variety of human thought
distilled to science of math
abstracted to logics
implemented in
software
TP



=====
Mechanised Foundation of Math

Conclusion

Renewed Pedagogy for the Future



STEM

individualise,
meaningful math
inquiry-based learning
embodiment, social experience
Effective Practice in Math Education

Let's hope for **joint inspiration** how to relate the sides !

Thank you for listening to this invitation !

infinite variety of human thought
distilled to science of math
abstracted to logics
implemented in
software
TP



=====
Mechanised Foundation of Math

Conclusion

Renewed Pedagogy for the Future



STEM
individualise,
meaningful math
inquiry-based learning
embodiment, social experience
Effective Practice in Math Education

Let's hope for **joint inspiration** how to relate the sides !

Thank you for listening to this invitation !

infinite variety of human thought
distilled to science of math
abstracted to logics
implemented in
software
TP



=====
Mechanised Foundation of Math

Conclusion

Renewed Pedagogy for the Future



STEM
individualise,
meaningful math
inquiry-based learning
embodiment, social experience
Effective Practice in Math Education

Let's hope for **joint inspiration** how to relate the sides !

Thank you for listening to this invitation !

infinite variety of human thought
distilled to science of math
abstracted to logics
implemented in
software
TP



Mechanised Foundation of Math

Conclusion

Renewed Pedagogy for the Future



STEM
individualise,
meaningful math
inquiry-based learning
embodiment, social experience
Effective Practice in Math Education

Let's hope for joint inspiration how to relate the sides !

Thank you for listening to this invitation !

Conclusion

Renewed Pedagogy for the Future



STEM

individualise,

meaningful math

inquiry-based learning

embodiment, social experience

Effective Practice in Math Education

infinite variety of human thought
distilled to science of math
abstracted to logics
implemented in
software
TP



Mechanised Foundation of Math

Let's hope for joint inspiration how to relate the sides !

Thank you for listening to this invitation !

Conclusion

Renewed Pedagogy for the Future

=====



STEM

individualise,
meaningful math
inquiry-based learning
embodiment, social experience
Effective Practice in Math Education

infinite variety of human thought
distilled to science of math
abstracted to logics
implemented in
software
TP



=====

Mechanised Foundation of Math

Let's hope for **joint inspiration** how to relate the sides !

Thank you for listening to this invitation !

Conclusion

Renewed Pedagogy for the Future

=====



STEM

individualise,
meaningful math
inquiry-based learning
embodiment, social experience
Effective Practice in Math Education

infinite variety of human thought
distilled to science of math
abstracted to logics
implemented in
software
TP



=====

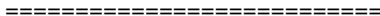
Mechanised Foundation of Math

Let's hope for joint inspiration how to relate the sides !

Thank you for listening to this invitation !

Conclusion

Renewed Pedagogy for the Future



STEM

individualise,

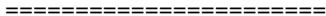
meaningful math

inquiry-based learning

embodiment, social experience

Effective Practice in Math Education

infinite variety of human thought
distilled to science of math
abstracted to logics
implemented in
software
TP



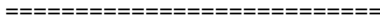
Mechanised Foundation of Math

Let's hope for joint inspiration how to relate the sides !

Thank you for listening to this invitation !

Conclusion

Renewed Pedagogy for the Future



STEM

individualise,

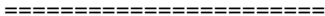
meaningful math

inquiry-based learning

embodiment, social experience

Effective Practice in Math Education

infinite variety of human thought
distilled to science of math
abstracted to logics
implemented in
software
TP



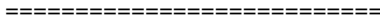
Mechanised Foundation of Math

Let's hope for **joint inspiration** how to relate the sides !

Thank you for listening to this invitation !

Conclusion

Renewed Pedagogy for the Future



STEM

individualise,

meaningful math

inquiry-based learning

embodiment, social experience

Effective Practice in Math Education

infinite variety of human thought
distilled to science of math
abstracted to logics
implemented in
software
TP



Mechanised Foundation of Math

Let's hope for **joint inspiration** how to relate the sides !

Thank you for listening to this invitation !

TP Technology I



P. Quaresma and R.-J. Back, editors,
Proceedings First Workshop on *TP Components for
Educational Software* .

Electronic Proceedings in Theoretical Computer
Science, Vol. 79, 2012.



Theorem Prover Isabelle's theories,
<http://isabelle.in.tum.de/dist/library/HOL/index.html>



Theorem Prover Coq,
<http://coq.inria.fr/>